WEATHER FORECAST APPLICATION

MINI PROJECT REPORT

submitted

in the partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

P. SOWMITH REDDY (20B81A05P0)

M. VIGNESH REDDY (20B81A05P9)

V. SUJITH REDDY (20PQ1A0567)

Under the guidance of

Mr.Syed Muqthadar Ali

Sr.Assistant Professor

CSE DEPARTMENT

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CVR COLLEGE OF ENGINEERING

(*An Autonomous institution, NBA, NAAC Accredited and Affiliated to JNTUH, Hyderabad*)

Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),
Rangareddy (D), Telangana- 501 510

May 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Project Report entitled “**WEATHER FORECAST APPLICATION**” being submitted by **P. SOWMITH REDDY (20B81A05P0), M. VIGNESH REDDY (20B81A05P9), V. SUJITH REDDY (20PQ1A0567)** in partial fulfillment for the award of Bachelor of Technology in Computer Science and Engineering to the CVR College of Engineering, to the CVR College of Engineering, is a record of bonafide work carried out by them under my Guidance and supervision during the year 2022-2023.

The results embodied in this project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Signature of the project guide	Signature of the HOD
Mr. Syed Muqthadar Ali	**Dr. A Vani Vathsala**
Sr. Assistant Professor	Department of CSE

Signature of the project coordinator	**External Examiner**
Dr. Venkatesh Sharma	
CSE Department	

DECLARATION

We hereby declare that the project report entitled "**Weather Forecast Application**" is an original work done and submitted to CSE Department, CVR College of Engineering, affiliated to Jawaharlal Nehru Technological University Hyderabad, in partial fulfillment of the requirement for the award of Bachelor of Technology in **Computer Science and Engineering** and it is a record of bonafide project work carried out by us under the guidance **Mr. Syed Muqthadar Ali Sr. Assistant Professor, Department of Computer Science and Engineering.**

We further declare that the work reported in this project has not been submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other Institute or University.

Signature of the Student
(P. SOWMITH REDDY)
(20B81A05P0)

Signature of the Student
(M. VIGNESH REDDY)
(20B81A05P9)

Signature of the Student
(V. SUJITH REDDY)
(20PQ1A0567)

ACKNOWLEDGEMENT

The satisfaction of completing this project would be incomplete without mentioning our gratitude towards all the people who have supported us. Constant guidance and encouragement have been instrumental in the completion of this project.

First and foremost, we thank our Principal **Prof K. Ramamohan Reddy**, Vice Principal **Prof. L. C. Siva Reddy**, Head of Department **Dr. A. Vani Vathsala** for availing infrastructural facilities to complete the mini project in time.

We offer our sincere gratitude to our internal guide **Mr. Syed Muqthadar Ali,** Senior Professor, CSE Department, CVR College of Engineering for his immense support, timely co-operation, and valuable advice throughout the course of our project work.

We would like to thank the Professor In-Charge of Projects, **Dr. Venkatesh Sharma**, Professor, Information Technology for his valuable suggestions in implementing the project.

We are thankful to **K. Giri Babu**, Project Coordinator, CSE Department, CVR College of Engineering for his supportive guidelines and for having provided the necessary help for carrying forward this project without any obstacles and hindrances.

We are thankful to the management for their supportive guidelines and for having provided the necessary help for carrying forward this project without any obstacles and hindrances.

P. SOWMITH REDDY (20B81A05P0)
M. VIGNESH REDDY (20B81A05P9)
V. SUJITH REDDY (20PQ1A0567)

ABSTRACT

This project involves the development of a weather forecast application using an API (Application Programming Interface) to access real-time weather data. The aim of this application is to provide users with accurate weather information that they can use to plan their daily activities.

The application will use a Restful API to retrieve weather data from a third-party weather service provider. The API will provide data such as current weather conditions, temperature, humidity, wind speed, and precipitation. The application will then display this information to the user in an easy-to-understand format, such as charts, graphs, or maps.

The application will also allow users to input their location and receive localized weather data. This will be accomplished by using the user's GPS coordinates to retrieve weather data from the API.

Additionally, the application will feature advanced features such as push notifications to alert users of significant weather changes, the ability to save and track multiple locations, and the option to view historical weather data.

Overall, the weather forecast application will provide users with valuable information that they can use to plan their daily activities and stay informed about weather conditions in their area.

LIST OF FIGURES

TABLE OF CONTENTS

1. INTRODUCTION

Weather forecasting is an essential aspect of our daily lives. It helps us plan our activities and make informed decisions. With the advent of technology, weather forecast applications have become increasingly popular. In this presentation, we will discuss a weather forecast application developed using ReactUS.
ReactJS is a popular JavaScript library used for building user interfaces. It allows developers to create reusable UI components and has gained popularity due to its
simplicity and efficiency.

A weather forecast application built using ReactJS can be a useful tool for users who want to stay informed about weather conditions in their local area or in other locations they are interested in. In this application, users can search for a specific location and get a detailed weather forecast for the next few days.

To build a weather forecast application using ReactJS, you will need to integrate with a weather API that provides the necessary data. Some popular options include OpenWeatherMap, WeatherStack, and AccuWeather. Once you have access to the API, you can start building the user interface using React components.

1.1 PROBLEM STATEMENT:

Developing a weather forecast application is a complex task that requires integrating data from various sources and providing users with accurate and reliable weather conditions. The challenge lies in providing users with up-to-date weather information that is tailored to their specific location and preferences. Additionally, users may have different needs and interests when it comes to weather, such as planning outdoor activities, travel, or work-related tasks. Therefore, the problem statement for a weather forecast application is to develop a user-friendly and reliable application that can provide personalized weather forecast information to help users plan and prepare for current weather conditions. The application should also provide users weather events and should be available on desktop, to ensure easy accessibility

Currently, there is a lack of user-friendly weather forecast applications that provide accurate and up-to-date information to users. Existing weather applications often have complex interfaces, provide limited information, or require users to navigate through multiple pages to access the information they need. Furthermore, some applications do not provide data for smaller regions or cities, making it difficult for users in those areas to plan their activities accordingly.

To address these issues, we aim to develop a weather forecast application using React.js that provides a user-friendly interface and accurate weather information. The application will utilize external APIs, such as OpenWeatherMap and Dark Sky, to fetch real-time weather data and forecasts. It will also allow users to customize their preferences, such as temperature units, language, and themes, to personalize their experience.

1.2 OBJECTIVE:

Provide real-time weather information: The primary objective of a weather forecast application built using ReactJS is to provide users with up-to-date and accurate weather information for their current location and other locations of interest.

Create an intuitive user interface: Another objective of a weather forecast application built using ReactJS is to create a user-friendly and accessible interface that is easy for users to navigate and understand.

Personalize the user experience: A weather forecast application built using ReactJS can aim to provide a personalized experience for users, such as customized weather alerts and the ability to save frequently searched locations.

Optimize for mobile use: With the majority of internet usage occurring on mobile devices, a weather forecast application built using ReactJS should be fully responsive and optimized for mobile use.

Improve accessibility: Accessibility should also be a key objective when building a weather forecast application using ReactJS, to ensure that all users, regardless of their abilities, can access and use the application.

Integrate with other applications: To provide a more comprehensive weather forecasting tool, a weather forecast application built using ReactJS can aim to integrate with other applications, such as travel planning or outdoor activity apps, to provide users with more contextualized weather information.

1.3 MOTIVATION:

1.Growing demand: Weather forecasting is a fundamental need for many people, from planning outdoor activities to making travel plans. By building a weather forecast application using ReactJS, you are meeting a growing demand for real-time weather data and providing a valuable service to your users.

2.**Learning opportunities**: Building a weather forecast application using ReactJS is a great way to learn and practice front-end development skills. You will have the opportunity to work with React components, state management, API integration, and user interface design.

3.Customization options: With a weather forecast application built using ReactJS, you have the flexibility to customize the user interface and features to meet the needs of your target audience. You can include options for different units of measurement, save frequently searched locations, and provide personalized weather alerts.

4.Accessibility: By following accessibility guidelines when building your weather forecast application using ReactJS, you are making your application accessible to all users, regardless of their abilities. This is an important consideration that can improve the user experience and reach a wider audience.

5.Integration with other applications: A weather forecast application built using ReactJS can be integrated with other applications, such as travel planning or outdoor activity apps. This can provide a seamless user experience and improve the overall value of your application.

Overall, building a weather forecast application using ReactJS can be a motivating and rewarding experience that provides a valuable service to users. It also provides an opportunity to learn and practice front-end development skills and contribute to the growing demand for real-time weather data.

1.4 PROJECT REPORT ORGANIZATION:

The project is organized into several key components, each with specific roles and responsibilities. These include:

1. User Interface: This component is responsible for providing an intuitive and easy-to-use interface for users to interact with the application. The interface consists of a search bar where users can input their location, and a display area where weather data is presented.

2. Data Retrieval: This component is responsible for retrieving weather data from the OpenWeather API. The API provides real-time weather data for a given location, including temperature, humidity ,pressure.

3. Data Processing: This component is responsible for processing the data retrieved from the API and presenting it in a user-friendly format. The data is formatted to include the current weather conditions.

4. Error Handling: This component is responsible for handling errors that may occur during the data retrieval and processing phases. The application is designed to handle common errors, such as invalid location input or API errors.

5. Deployment: This component is responsible for deploying the application to a web server, making it accessible to users via the internet, ensuring high availability and scalability.

2. LITERATURE SURVEY

1. "Advances in Weather and Climate Forecasting" by Kevin E. Trenberth et al. This review article discusses the advancements in weather and climate forecasting, including the use of satellite and radar data, ensemble forecasting, and data assimilation techniques. The authors also highlight the challenges and limitations of weather forecasting, such as the need for high-resolution models and the difficulty of predicting extreme weather events.

2. "Weather Forecasting: A Review of Forecast Verification and Evaluation Methods" by Tilmann Gneiting et al. This review article provides an overview of the various methods used to verify and evaluate weather forecasts, including probabilistic and deterministic approaches. The authors also discuss the importance of proper evaluation methods in improving the accuracy and reliability of weather forecasts.

3. "The Importance of Weather Forecasting for Agriculture: A Review" by Cristine L. S. Morgan et al. This review article discusses the role of weather forecasting in agriculture, including the use of weather information for crop management, pest control, and irrigation scheduling. The authors also highlight the challenges and limitations of weather forecasting in agriculture, such as the need for more localized and accurate weather predictions.

4. "Deep Learning for Weather Forecasting: A Review" by Xingang Fan et al. This review article discusses the use of deep learning techniques in weather forecasting, including convolutional neural networks and recurrent neural networks. The authors highlight the potential of deep learning in improving the accuracy and efficiency of weather forecasting, especially for short-term predictions.

5."Evaluation of Weather Forecasting Services: A Review" by Yongwei Sheng et al. This review article provides an overview of the various methods used to evaluate weather forecasting services, including user surveys, performance metrics, and cost-benefit analyses. The authors also discuss the challenges and opportunities of evaluating weather forecasting services, such as the need for standardized evaluation methods and the potential for incorporating user feedback into the evaluation process.

2.1 SURVEY OF MAJOR AREA RELATED TO PROJECT:

1. Meteorology: This is the study of atmospheric processes and phenomena, including temperature, pressure and humidity. Meteorology provides the scientific basis for weather forecasting by analyzing weather patterns and identifying trends.

2. Data Collection: Weather forecast applications rely on a vast amount of data collected from various sources, including weather stations, satellites, radars, and weather balloons. The collection and processing of this data are critical to providing accurate and timely weather predictions.

3. Data Analysis: Advanced statistical and mathematical models are used to analyze weather data and identify patterns and trends. This analysis helps forecasters develop predictive models for future weather events.

4. Machine Learning and Artificial Intelligence: Weather forecast applications can leverage machine learning and artificial intelligence algorithms to improve the accuracy and reliability of weather predictions. These algorithms can analyze large amounts of data and identify complex patterns that may not be apparent to human forecasters.

5. User Interface and Experience: Weather forecast applications must provide a user-friendly and accessible interface that enables users to quickly and easily access weather information. The user interface should provide relevant and personalized information that is easy to understand and interpret.

6. Infrastructure: Weather forecast applications require robust and reliable infrastructure, including servers, databases, and networking equipment, to store and process weather data and provide real-time information to users.

7. Emergency Management: Weather forecast applications play a critical role in emergency management, providing real-time weather information to first responders and emergency managers. These applications can help identify the impact of severe weather events on people and infrastructure.

2.2 EXSISTING WORK:

There are many existing weather forecast web applications that have been developed using various technologies, including ReactJS. Here are a few examples:

1. AccuWeather - AccuWeather is a popular weather forecasting application that provides users with real-time weather information and alerts for their current location and other locations around the world. The application is built using ReactJS and other web technologies.

2. Weather Underground - Weather Underground is another popular weather forecasting application that uses ReactJS and other web technologies to provide users with accurate and up-to-date weather information and alerts.

3. Dark Sky - Dark Sky is a weather forecasting application that provides users with hyper-local weather information and forecasts. The application is built using ReactJS and other web technologies.

4. Weather.com - Weather.com is a weather forecasting application that uses ReactJS and other web technologies to provide users with up-to-date weather information and alerts for their current location and other locations around the world.

5. OpenWeatherMap - OpenWeatherMap is a weather forecasting application that uses ReactJS and other web technologies to provide users with real-time weather information and forecasts for their current location and other locations around the world.

These are just a few examples of the many weather forecast web applications that have been developed using ReactJS and other web technologies.

2.3 LIMITATIONS OF EXISTING WORK:

While existing weather forecast applications provide valuable features and functionality, there are several limitations to consider. Some of the limitations of existing work in the weather forecast application domain are:

1. Limited Accuracy: Weather forecast applications use weather data provided by third-party sources such as National Weather Service or weather stations, which may not always be accurate for specific locations. The accuracy of weather forecasts can be affected by several factors such as the quality of data, the complexity of weather patterns, and the limited availability of data in certain locations.

2. Limited Availability: Many weather forecast applications are restricted to specific geographic regions or countries. This limitation can be a challenge for users who travel to different parts of the world or live in remote locations with limited data availability.

3. Limited Customization: Weather forecast applications offer limited customization options for users. This can be a limitation for users who require specific weather data or alerts that are not available in the application.

4. Inaccurate or Delayed Alerts: Weather forecast applications may not always provide timely or accurate weather alerts, which can be a significant limitation for users who rely on such alerts for their safety and well-being.

5. Limited Integration: Many weather forecast applications do not integrate with other applications or devices, limiting their usefulness and functionality for users who require weather data in their day-to-day activities.

3. REQUIREMENT SPECIFICATIONS

3.1 SOFTWARE REQUIREMENTS:

1.Text editor or integrated development environment (IDE): Developers can use a text editor or IDE such as VisualStudioCode, Atom, or WebStorm to write and edit the code for the application.

2. ReactJS: The application requires the ReactJS library to develop the front-end components and render the user interface.

3. Node.js: Node.js is a JavaScript runtime environment that allows developers to build server-side applications using JavaScript. It may be used in the backend to handle data processing and database operations**.**

4. APIs: The application may require access to weather APIs such as OpenWeatherMap, Weather
Underground, or Dark Sky to fetch realtime weather data.
5. Database: If the application needs to store and manage large amounts of weather data, a database such as MySQL, PostgreSQL, or MongoDB may be needed.

6. Deployment tools: The application needs to be deployed on a server to make it available to users. Deployment tools such as Heroku, AWS Elastic Beanstalk, or Digital Ocean may be used to deploy and manage the application!

3.2 HARDWARE REQUIREMENTS :

1. Processor: The application can run on a processor with a clock speed of at least 2 GHz or higher.

2. Memory (RAM): The application requires a minimum of 4 GB RAM for optimal performance. However, depending on the size of the application and the expected traffic, more RAM may be needed.

3. Storage: The application requires storage space to store the application code, database, and other files. A minimum of 50 GB storage space is recommended.

4. Network: The application requires a stable internet connection with a minimum bandwidth of 5 Mbps for optimal performance.

5. Operating System: The application can run on any operating system that supports Node.js and React, such as Windows, macOS, and Linux.

6. Browser: The application can be accessed using any modern web browser, such as Google Chrome, Mozilla Firefox, or Safari.

7. Cloud Hosting: If the application is deployed on a cloud platform, such as Heroku or AWS, the hardware requirements would depend on the pricing plan chosen. For example, a higher pricing plan may provide more **CPU, RAM, and storage resources for the application.**

3.3 DESIGN REQUIREMENTS:

- Use Node.js and React to build the application.
- Use APIs to fetch weather data and forecasts.

3.4 ACCEPTANCE CRITERIA:

- The application should be able to provide accurate weather data and forecasts.
- The application should be user-friendly and easy to navigate.
- The application should be compatible with different operating systems.

4. PROPOSED SYSTEM DESIGN:

4.1 System Architecture:

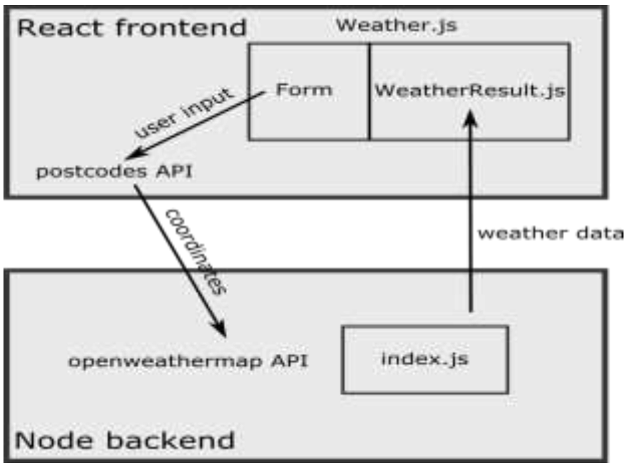


Figure 1 System Architecture

1. Client-Side: The client-side of the application would be developed using React, which is responsible for rendering the user interface and handling user interactions. The client-side would communicate with the server-side via RESTful APIs to fetch weather data and other information.

2. Server-Side: The server-side of the application would be developed using Node.js, which would provide a platform for building a scalable and efficient backend system. The server-side would communicate with external APIs, such as OpenWeatherMap and Dark Sky, to fetch real-time weather data and forecasts.

3. API Gateway: The API Gateway would act as a proxy server that routes incoming requests from the client-side to the appropriate backend services. It would also be responsible for handling authentication and authorization to ensure that only authorized users can access the application.

4. Database: The database would store user preferences, such as temperature units, language, and themes, which can be used to customize the user experience.

5. External APIs: The application would use external APIs, such as OpenWeatherMap and Dark Sky, to fetch weather data and forecasts. These APIs would be accessed via HTTP requests, and the data would be processed and formatted before being sent to the client-side.

6. Cache: The application can use a caching mechanism, such as Redis, to store frequently accessed data and reduce the response time for subsequent requests.

4.2 UI DESIGN:

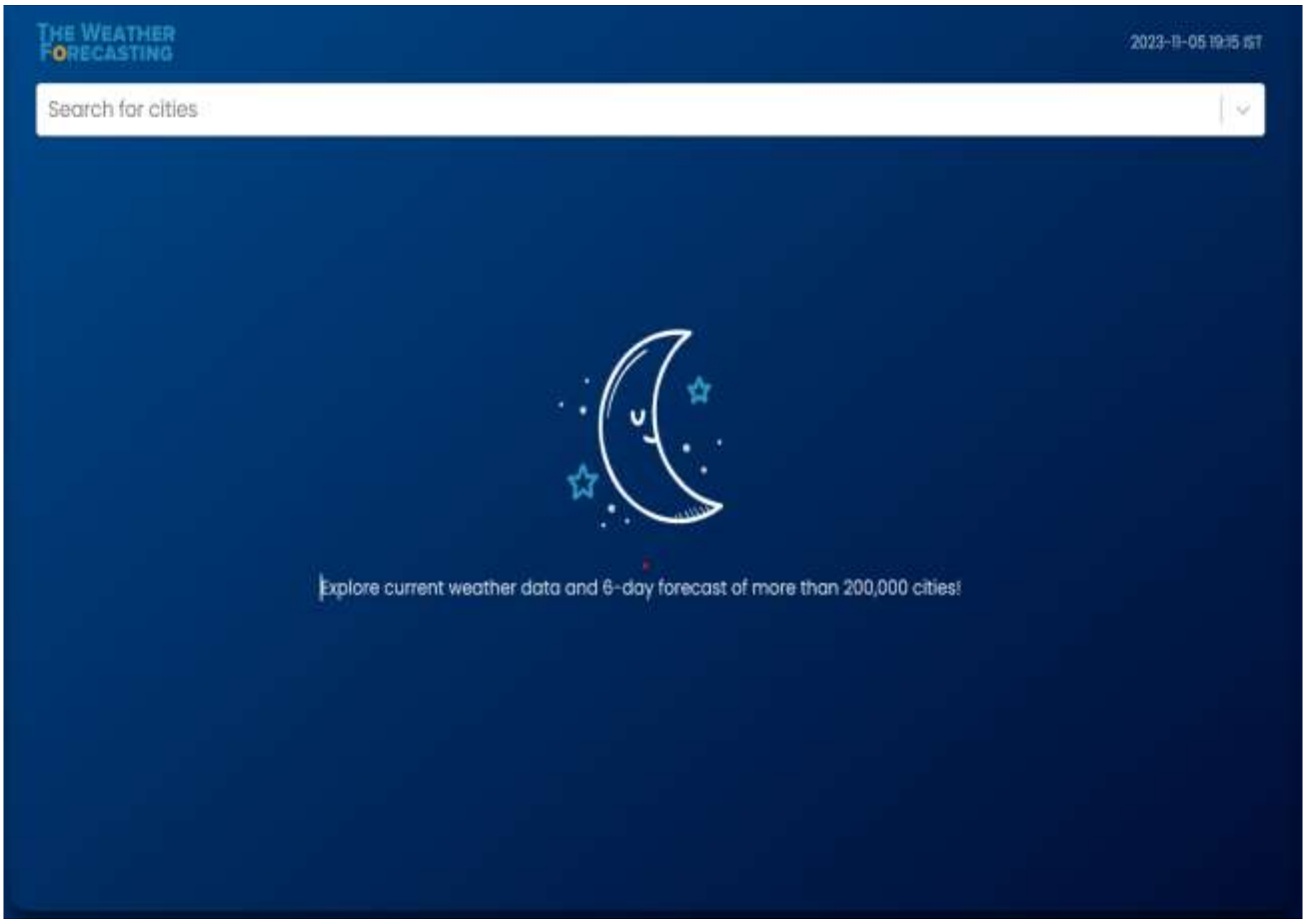

THE WEATHER FORECASTING
2023-11-05 19:15 IST
Search for cities
Explore current weather data and 6-day forecast of more than 200,000 cities!

Figure 2 UI Design

5. UML Diagrams:

5.1Use Case Diagram

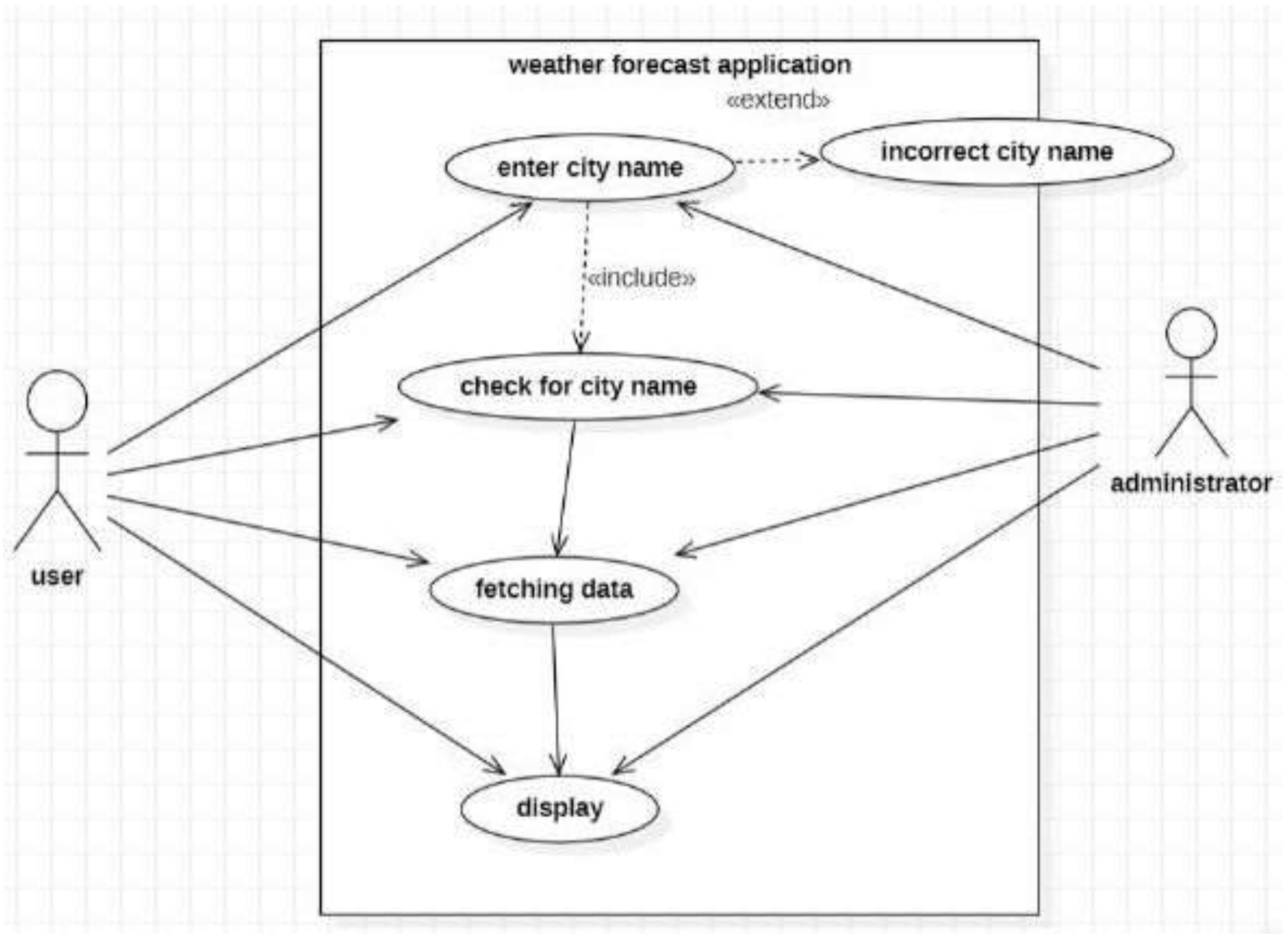


Figure 3 Use Case Diagram

5.2 Component Diagram:

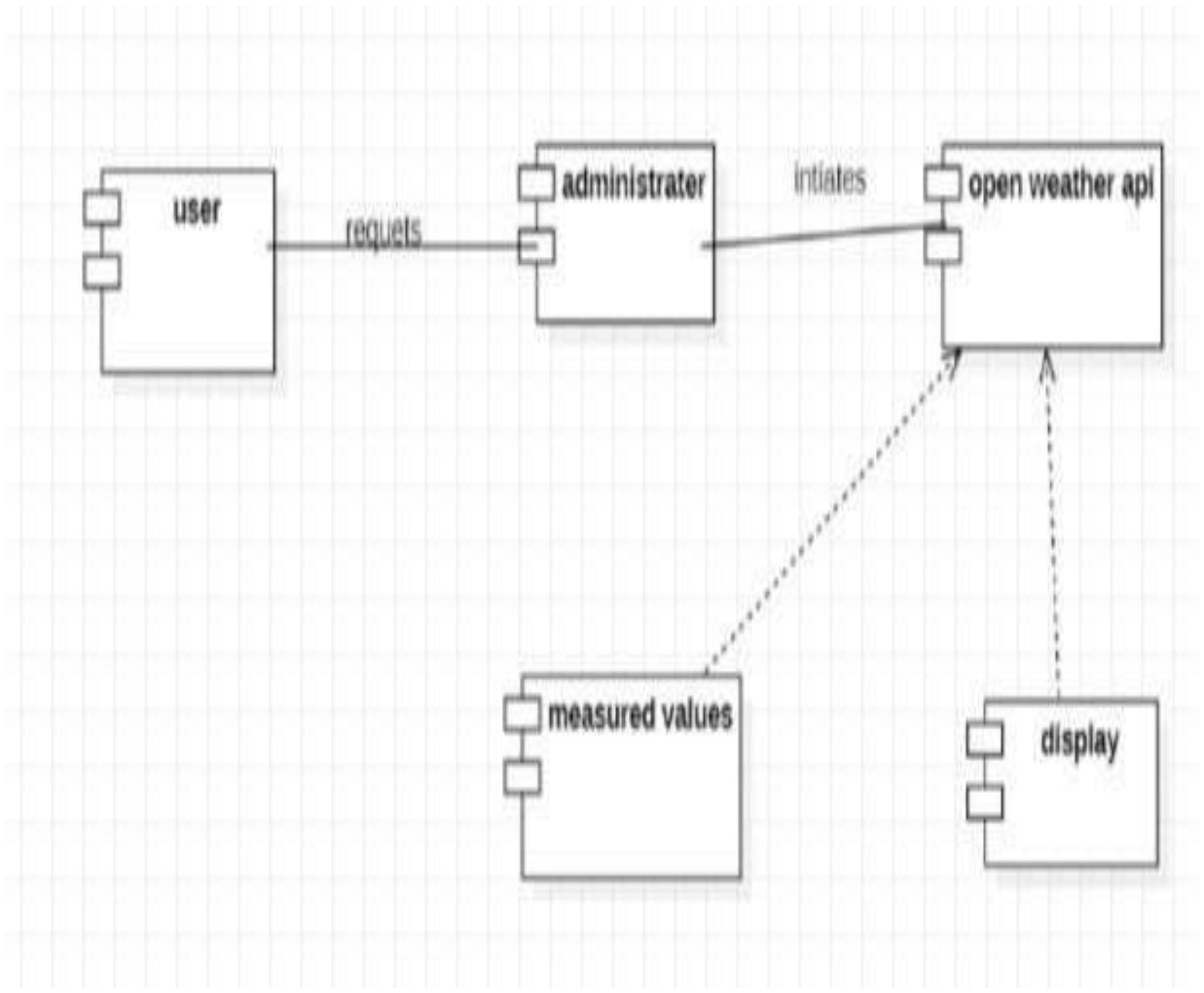


Figure 4 Component Diagram

5.3 Deployment Diagram:

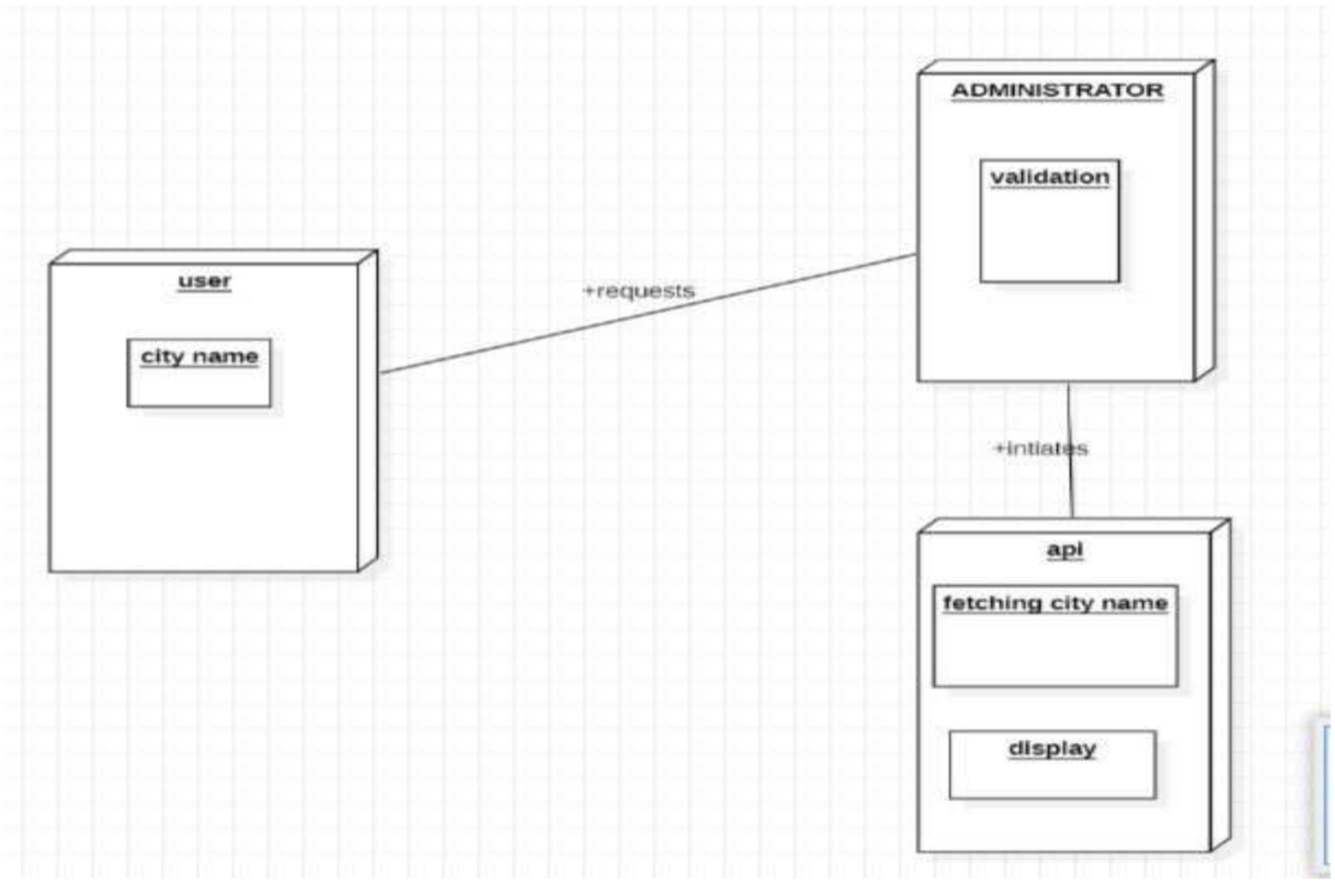


Figure 5 Deployment Diagram

6. IMPLEMENTATION

6.1 USER INTERFACE DESIGN:

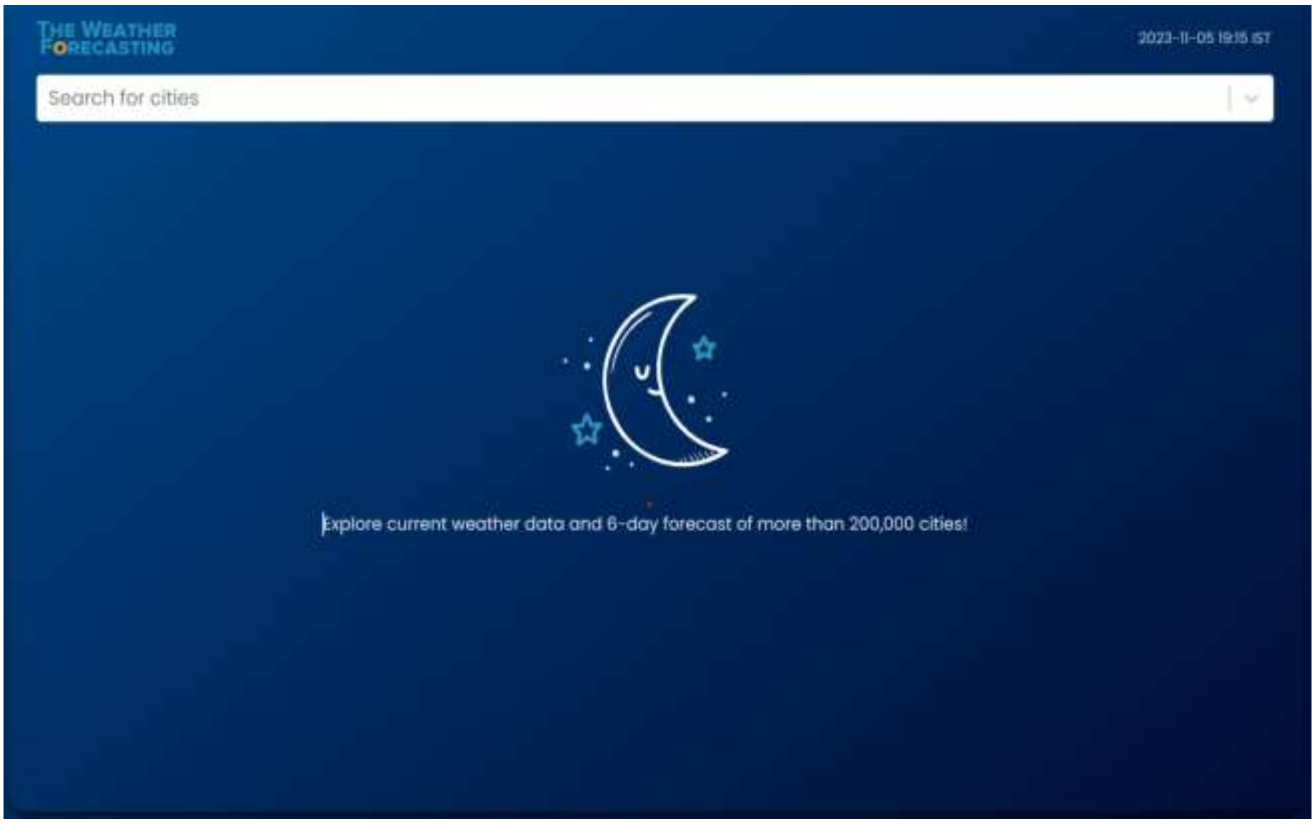
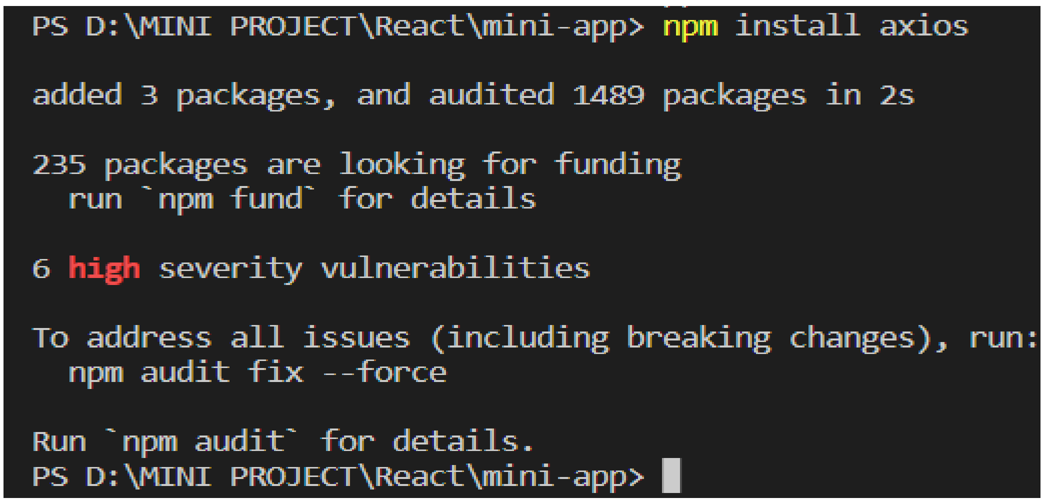

THE WEATHER FORECASTING
2023-11-05 19:15 IST
Search for cities
Explore current weather data and 6-day forecast of more than 200,000 cities!

Figure 6 User Interface Design

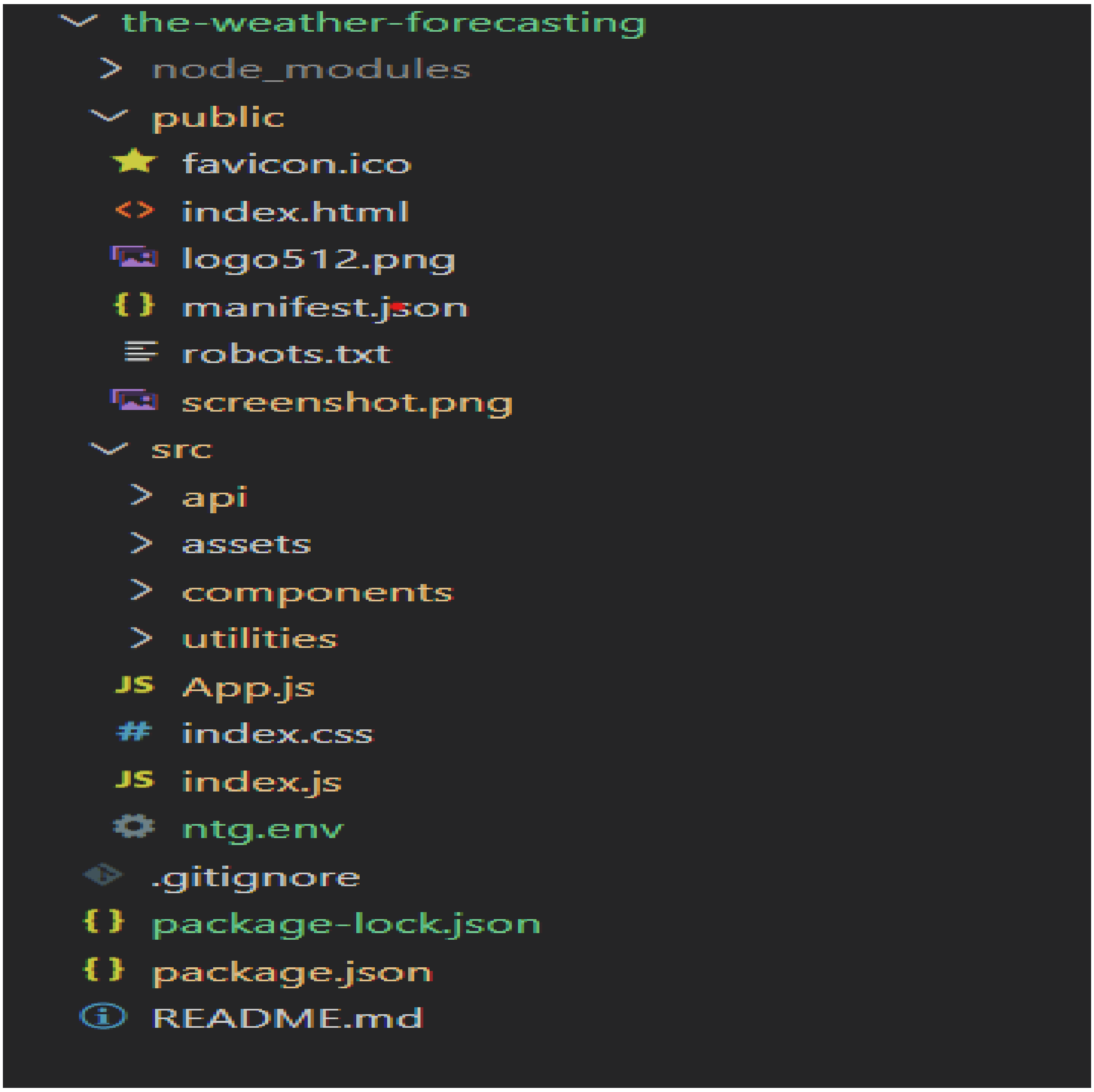
6.2 PACKAGE INSTALLED:



```
PS D:\MINI PROJECT\React\mini-app> npm install axios

added 3 packages, and audited 1489 packages in 2s

235 packages are looking for funding
  run `npm fund` for details

6 high severity vulnerabilities

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
PS D:\MINI PROJECT\React\mini-app>
```

Figure 7 Package Installed

6.3 FILE STRUCTURE:



```
the-weather-forecasting
  node_modules
  public
    favicon.ico
    index.html
    logo512.png
    manifest.json
    robots.txt
    screenshot.png
  src
    api
    assets
    components
    utilities
    App.js
    index.css
    index.js
    ntg.env
  .gitignore
  package-lock.json
  package.json
  README.md
```

Figure 8 File Structure

6.3.1 ICONS USED:

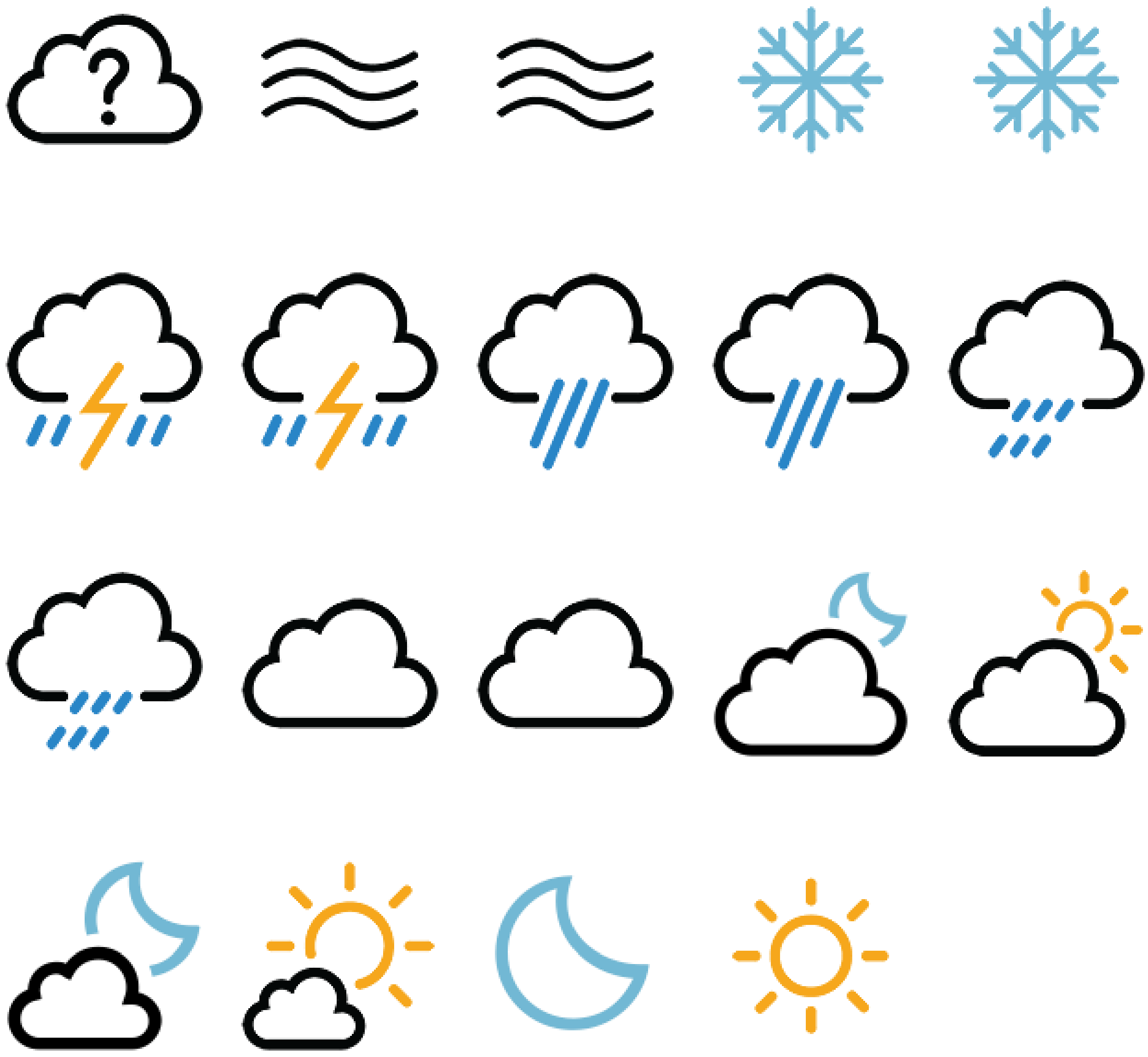


Figure 9 Icons Used

6.3.2 CODE:

OpenWeatherService.js

```
const GEO_API_URL = 'https://wft-geo-db.p.rapidapi.com/v1/geo';

const WEATHER_API_URL = 'https://api.openweathermap.org/data/2.5';
const WEATHER_API_KEY = '9a652326a39f0a5e06b79aa69c30dbc7';

const GEO_API_OPTIONS = {
  method: 'GET',
  headers: {
    'X-RapidAPI-Key': '4f0dcce84bmshac9e329bd55fd14p17ec6fjsnff18c2e61917',
    'X-RapidAPI-Host': 'wft-geo-db.p.rapidapi.com',
  },
};

export async function fetchWeatherData(lat, lon) {
  try {
    let [weatherPromise, forcastPromise] = await Promise.all([
      fetch(

`${WEATHER_API_URL}/weather?lat=${lat}&lon=${lon}&appid=${WEATHER_API_KEY}&units=
metric`
      ),
      fetch(

`${WEATHER_API_URL}/forecast?lat=${lat}&lon=${lon}&appid=${WEATHER_API_KEY}&units=
metric`
      ),
    ]);

    const weatherResponse = await weatherPromise.json();
    const forcastResponse = await forcastPromise.json();
    return [weatherResponse, forcastResponse];
  } catch (error) {
    console.log(error);
  }
}

export async function fetchCities(input) {
  try {
    const response = await fetch(
      `${GEO_API_URL}/cities?minPopulation=10000&namePrefix=${input}`,
      GEO_API_OPTIONS
    );

    const data = await response.json();
    return data;
  } catch (error) {
    console.log(error);
    return;
  }
}
```

search.js

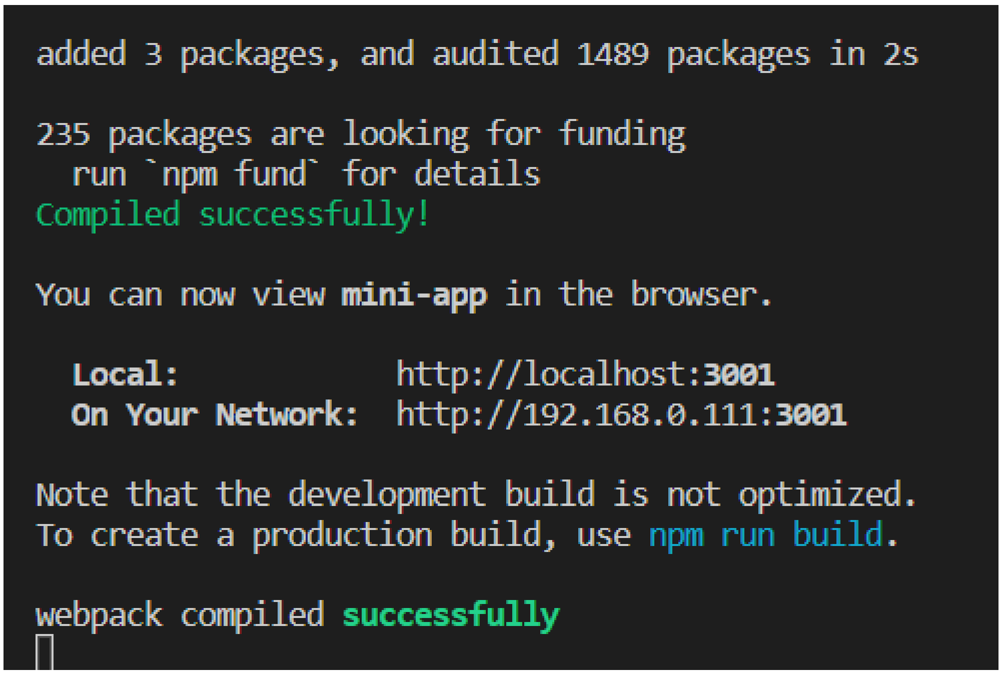
```
import React, { useState } from 'react';
import { AsyncPaginate } from 'react-select-async-paginate';
import { fetchCities } from '../../api/OpenWeatherService';

const Search = ({ onSearchChange }) => {
  const [searchValue, setSearchValue] = useState(null);

  const loadOptions = async (inputValue) => {
    const citiesList = await fetchCities(inputValue);

    return {
      options: citiesList.data.map((city) => {
        return {
          value: `${city.latitude} ${city.longitude}`,
          label: `${city.name}, ${city.countryCode}`,
        };
      }),
    };
  };
  const onChangeHandler = (enteredData) => {
    setSearchValue(enteredData);
    onSearchChange(enteredData);
  };
  return (
    <AsyncPaginate
      placeholder="Search for cities"
      debounceTimeout={600}
      value={searchValue}
      onChange={onChangeHandler}
      loadOptions={loadOptions}
    />
  );
};

export default Search;
```

UTCDateTime.js

```
import { Typography } from '@mui/material';
import React from 'react';
import { getUTCDatetime } from '../../utilities/DatetimeUtils';

const UTCDatetime = () => {
  const utcFullDate = getUTCDatetime();
  const utcTimeValue = (
    <Typography
      variant="h3"
      component="h3"
      sx={{
        fontWeight: '400',
        fontSize: { xs: '10px', sm: '12px' },
        color: 'rgba(255, 255, 255, .7)',
        lineHeight: 1,
        paddingRight: '2px',
        fontFamily: 'Poppins',
      }}
    >
      {utcFullDate} IST
    </Typography>
  );
  return utcTimeValue;
};

export default UTCDatetime;
```

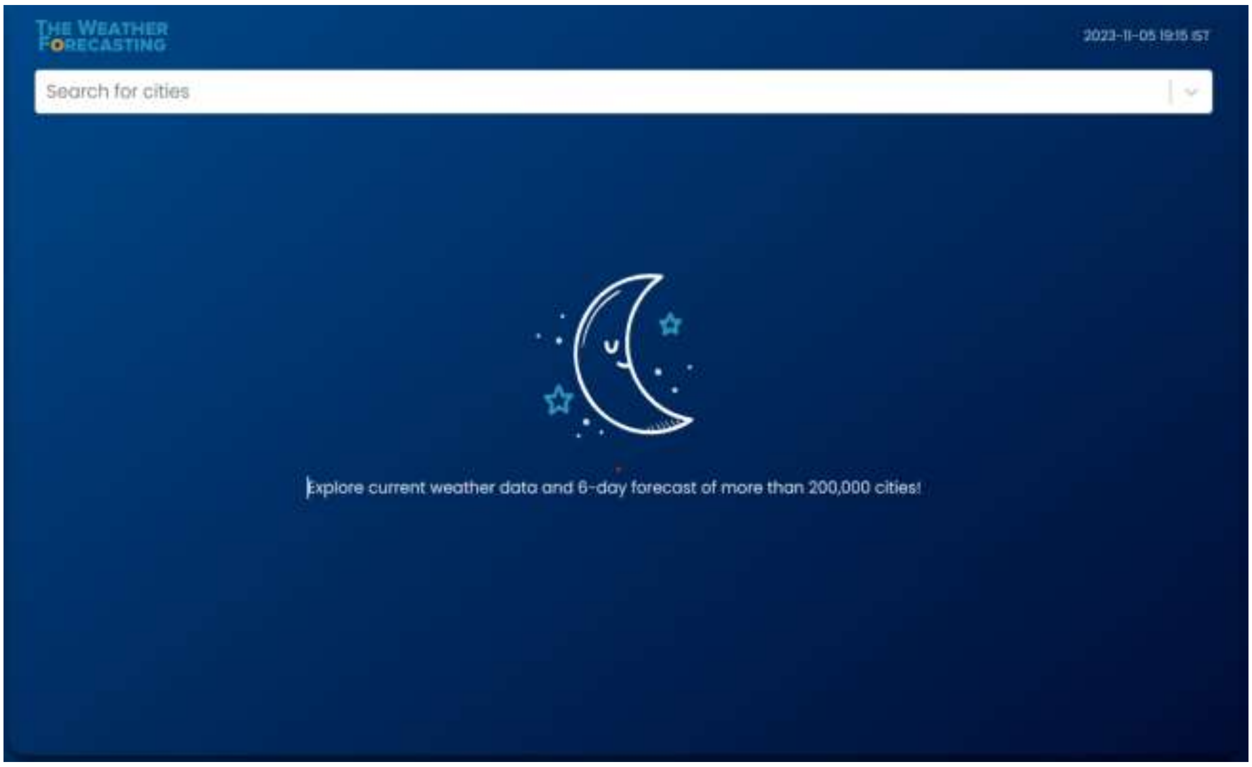
7. TESTING :

> > npx react-scripts start



```
added 3 packages, and audited 1489 packages in 2s

235 packages are looking for funding
  run `npm fund` for details
Compiled successfully!

You can now view mini-app in the browser.

  Local:            http://localhost:3001
  On Your Network:  http://192.168.0.111:3001

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

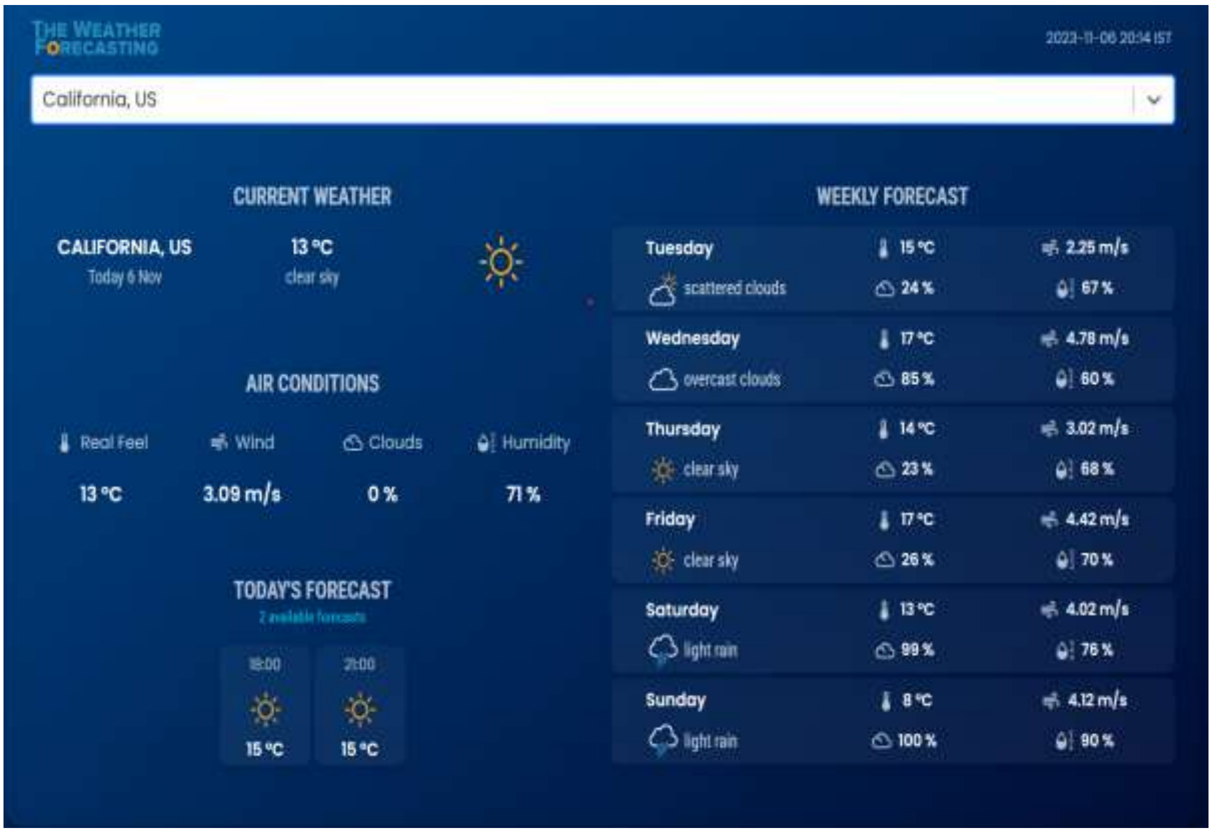
Figure 11 Compilation



Explore current weather data and 6-day forecast of more than 200,000 cities!

Figure 10 Execution

Results :


THE WEATHER FORECASTING

2023-11-06 20:14 IST

California, US

CURRENT WEATHER

CALIFORNIA, US
Today 6 Nov

13 °C
clear sky

AIR CONDITIONS

Real Feel	Wind	Clouds	Humidity
13 °C	3.09 m/s	0 %	71 %

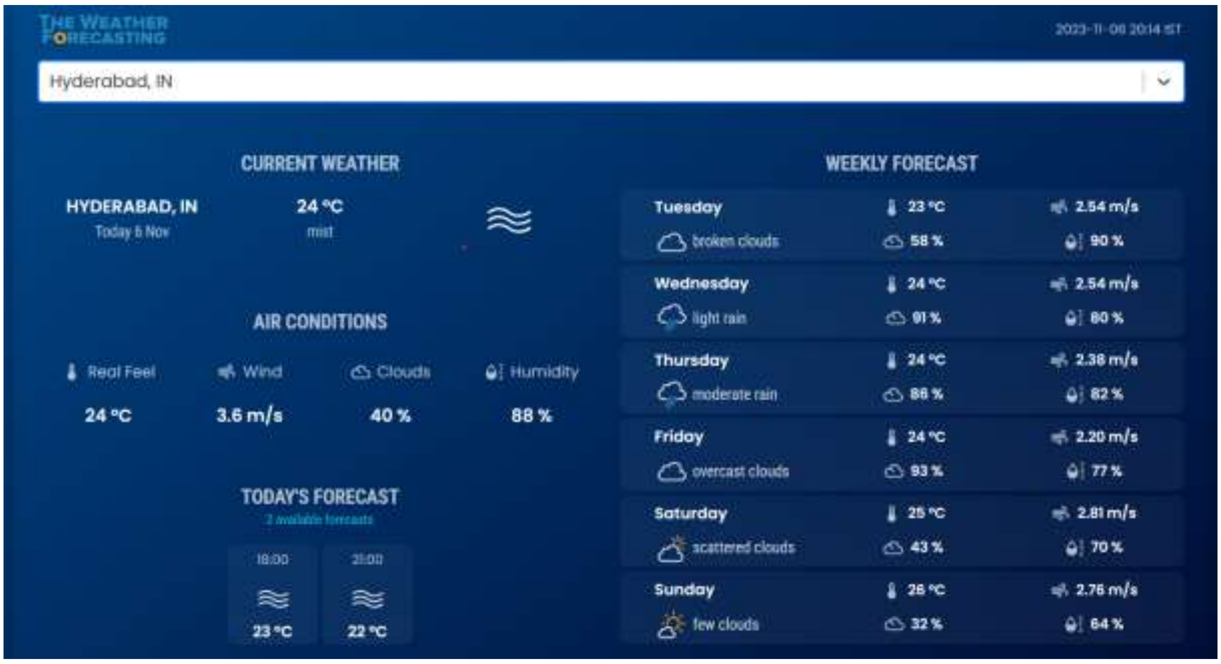
TODAY'S FORECAST
2 available forecasts

18:00	21:00
15 °C	15 °C

WEEKLY FORECAST

Tuesday	15 °C	2.25 m/s
scattered clouds	24 %	67 %
Wednesday	17 °C	4.78 m/s
overcast clouds	85 %	60 %
Thursday	14 °C	3.02 m/s
clear sky	23 %	68 %
Friday	17 °C	4.42 m/s
clear sky	26 %	70 %
Saturday	13 °C	4.02 m/s
light rain	99 %	76 %
Sunday	8 °C	4.12 m/s
light rain	100 %	90 %

Figure 12 Output 1


THE WEATHER FORECASTING

2023-11-06 20:14 IST

Hyderabad, IN

CURRENT WEATHER

HYDERABAD, IN
Today 6 Nov

24 °C
mist

AIR CONDITIONS

Real Feel	Wind	Clouds	Humidity
24 °C	3.6 m/s	40 %	88 %

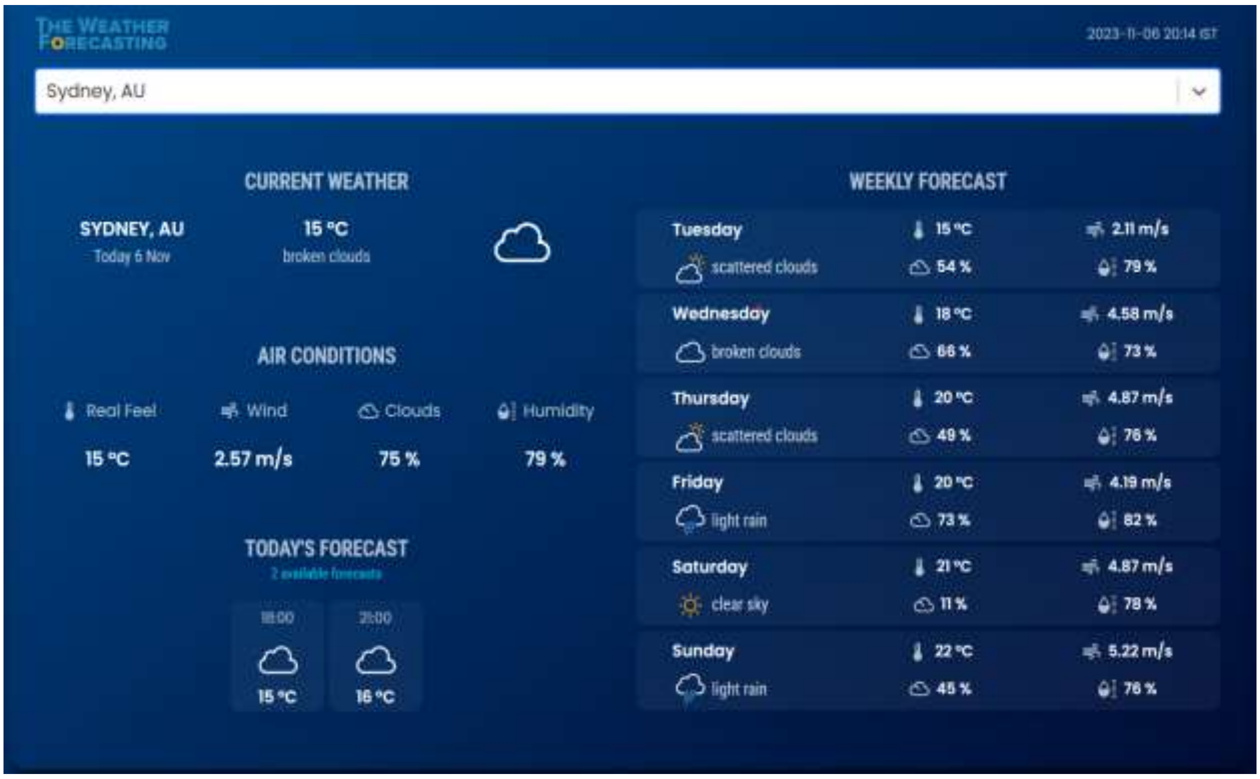
TODAY'S FORECAST
2 available forecasts

18:00	21:00
23 °C	22 °C

WEEKLY FORECAST

Tuesday	23 °C	2.54 m/s
broken clouds	58 %	90 %
Wednesday	24 °C	2.54 m/s
light rain	91 %	80 %
Thursday	24 °C	2.38 m/s
moderate rain	86 %	82 %
Friday	24 °C	2.20 m/s
overcast clouds	93 %	77 %
Saturday	25 °C	2.81 m/s
scattered clouds	43 %	70 %
Sunday	26 °C	2.76 m/s
few clouds	32 %	64 %

Figure 13 Output 2


THE WEATHER FORECASTING

2023-11-06 20:14 IST

Sydney, AU

CURRENT WEATHER

SYDNEY, AU
Today 6 Nov

15 °C
broken clouds

AIR CONDITIONS

Real Feel	Wind	Clouds	Humidity
15 °C	2.57 m/s	75 %	79 %

TODAY'S FORECAST
2 available forecasts

18:00	21:00
15 °C	16 °C

WEEKLY FORECAST

Tuesday	15 °C	2.11 m/s
scattered clouds	54 %	79 %
Wednesday	18 °C	4.58 m/s
broken clouds	66 %	73 %
Thursday	20 °C	4.87 m/s
scattered clouds	49 %	76 %
Friday	20 °C	4.19 m/s
light rain	73 %	82 %
Saturday	21 °C	4.87 m/s
clear sky	11 %	78 %
Sunday	22 °C	5.22 m/s
light rain	45 %	76 %

Figure 14 Output 3

CONCLUSION

In conclusion, a weather forecast application built using ReactJS can provide users with up-to-date weather information and forecasts for their current location and other locations around the world. The application can be designed using a modular structure, with each component responsible for specific functionality and displaying specific weather data to the user.

To build a weather forecast application using ReactJS, developers need to have a strong understanding of React's core concepts, such as components, props, and state, as well as web technologies such as APIs and data visualization libraries. Additionally, developers need to consider the user experience and design an intuitive user interface that makes it easy for users to search for and view weather information.

By following best practices in React development, such as using functional components and hooks, as well as implementing efficient data fetching and caching strategies, developers can build a robust and reliable weather forecast application that meets the needs of users.

Overall, a weather forecast application built using ReactJS can be a valuable tool for anyone looking to stay informed about weather conditions and make informed decisions based on current and future forecasts.

Future Enhancement

There are several potential future developments for a weather forecast application built using ReactJS, including:

1. Improving data visualization: Currently, most weather forecast applications display weather data using simple charts and graphs. In the future, developers could use more advanced data visualization techniques, such as heatmaps, animations, and interactive maps, to provide users with a more engaging and informative experience.

2. Integrating machine learning: Machine learning algorithms can be used to analyze weather data and make more accurate predictions about future weather patterns. By integrating machine learning capabilities into the application, developers could provide users with more precise and reliable weather forecasts.

3. Adding social features: Many users like to share weather-related information with their friends and family. In the future, developers could add social features to the application, such as the ability to share weather data on social media or create customized weather alerts for specific locations.

4. Supporting multiple languages: Weather forecast applications are used by people around the world, so it's important to support multiple languages. In the future, developers could add support for additional languages, making the application more accessible and user-friendly for people from different countries and cultures.

5. Providing personalized recommendations: By analyzing user behavior and weather data, developers could provide personalized recommendations for activities, such as outdoor sports, travel, or clothing. This would make the application even more useful for users and could help drive user engagement and retention.

Overall, there are many potential future developments for a weather forecast application built using ReactJS, and developers should continue to explore new technologies and strategies to improve the application's functionality and user experience.

REFERENCES

1. Joe Mayo, “**Microsoft Visual Studio 2010: A Beginner’s Guide**”, Tata McGraw Hill,2010.

2. Installation -NodeJS - Node.js (nodejs.org)

3. Obtained Weather Data from the OpenWeatherMap using API KEY provided by - Members (openweathermap.org)

4. Implemented the application using ReactHooks from - Built-in React Hooks – React

5. GeeksforGeeks - https://www.geeksforgeeks.org/

6. Wikipedia - https://www.wikipedia.org/

7. Learnt functions used through - React JSX (w3schools.com)